Université de Yaoundé I

Département d'Informatique

RAPPORT DE TP

INF231_EC2 : STRUCTURE DE DONNÉES II

Nom : ASSOUMOU YENE LAURENT KEVIN
Matricule : 24G2332
Niveau : L2
TP1 INF 231
Superviseur : Dr. MATATAGIA

Introduction

Ce rapport présente les différentes implémentations en langage C réalisées dans le cadre du TP1 du cours INF231_EC2 : Structure de Données II. L'objectif de ce travail est d'explorer et de manipuler les structures de données de base (tableaux, matrices, vecteurs) et de mettre en pratique des algorithmes fondamentaux tels que la somme et le produit de matrices, la recherche séquentielle, la vérification d'un tableau trié, le calcul de la médiane, l'inversion d'un tableau, ainsi que des opérations vectorielles et matricielles.

Table des matières

Exercices

Les exercices traités dans ce TP sont : - Somme de matrices - Produit de matrices - Recherche séquentielle dans un tableau - Multiplication a × b en utilisant exclusivement l’opérateur +1 - Test si un tableau est trié - Calcul de la médiane dans un tableau - Inversion d’un tableau - Produit vectoriel - Produit vecteur × matrice

Annexe : Code source complet en C

```c
#include <stdio.h>
#include <stdlib.h>

void sommeMat(int m, int n, int A[10][10], int B[10][10], int C[10][10]) {
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            C[i][j] = A[i][j] + B[i][j];
}

void prodMat(int m, int n, int p, int A[10][10], int B[10][10], int C[10][10]) {
    for (int i = 0; i < m; i++)
        for (int j = 0; j < p; j++) {
            C[i][j] = 0;
            for (int k = 0; k < n; k++)
                C[i][j] += A[i][k] * B[k][j];
        }
}

int recherche(int T[], int n, int x) {
    for (int i = 0; i < n; i++)
        if (T[i] == x) return i;
    return -1;
}

int mult(int a, int b) {
    int r = 0;
    for (int i = 0; i < b; i++)
        for (int j = 0; j < a; j++)
            r = r + 1;
    return r;
}

int trie(int T[], int n) {
    for (int i = 0; i < n-1; i++)
        if (T[i] > T[i+1]) return 0;
    return 1;
}

void trier(int T[], int n) {
    for (int i = 0; i < n-1; i++)
        for (int j = i+1; j < n; j++)
            if (T[i] > T[j]) {
                int tmp = T[i]; T[i] = T[j]; T[j] = tmp;
            }
}

float mediane(int T[], int n) {
    trier(T,n);
    if (n % 2 == 1) return T[n/2];
    return (T[n/2-1] + T[n/2]) / 2.0;
}

void inverser(int T[], int n) {
    for (int i = 0; i < n/2; i++) {
        int tmp = T[i];
        T[i] = T[n-1-i];
        T[n-1-i] = tmp;
    }
}

void prodVect(int A[3], int B[3], int C[3]) {
    C[0] = A[1]*B[2] - A[2]*B[1];
    C[1] = A[2]*B[0] - A[0]*B[2];
    C[2] = A[0]*B[1] - A[1]*B[0];
}

void vectMat(int n, int V[], int M[10][10], int R[]) {
    for (int j = 0; j < n; j++) {
        R[j] = 0;
        for (int i = 0; i < n; i++)
            R[j] += V[i] * M[i][j];
    }
}
```

```c
int main() {
    int choix;
    do {
        printf("\nMENU\n");
        printf("1. Somme de matrices\n");
        printf("2. Produit de matrices\n");
        printf("3. Recherche\n");
        printf("4. Multiplication a*b avec +1\n");
        printf("5. Tester tri\n");
        printf("6. Mediane\n");
        printf("7. Inverser tableau\n");
        printf("8. Produit vectoriel (3D)\n");
        printf("9. Vecteur x Matrice\n");
        printf("0. Quitter\n");
        printf("Choix: ");
        scanf("%d", &choix);
        // Implémentations des cas (voir code complet)...
    } while(choix!=0);
    return 0;
}
```

Conclusion

Ce TP a permis de mettre en pratique des notions essentielles liées aux structures de données et aux algorithmes en langage C. La réalisation des différentes opérations sur les tableaux, matrices et vecteurs a renforcé la compréhension de la manipulation des données en mémoire et de l’efficacité des algorithmes. Sur le plan académique, ces exercices constituent une base solide pour aborder des structures de données plus avancées comme les listes, piles, files et arbres. D’un point de vue personnel, ce travail m’a permis de développer une meilleure rigueur dans l’écriture du code, ainsi qu’une autonomie dans la recherche et la résolution de problèmes. Je retiens également l’importance de la structuration du code et de la clarté des algorithmes dans la compréhension et la maintenance d’un programme.